REMARKS

In the Office Action, the Examiner indicated that Claims 1, 5-10, 12-14, 16 and 17 are pending in the application and the Examiner rejected all claims.

The §112 Rejections

On page 2 of the Office Action, the Examiner rejected Claims 1, 5-10, 12-14, 16 and 17 under 35 U.S.C. §112, first paragraph, as failing to comply with the written description requirement. Applicant respectfully traverses this rejection. For the first time in the almost 7-year-and-counting prosecution of this application, the Examiner asserts that indexing located assets, which has been in the claims since the filing date of this application in 1999, was not described in the specification in such a way as to reasonably convey to one skilled in the relevant art that the inventor had possession of the invention at the time the invention was made.

Applicant describes in detail the indexing performed by the DI (Database and Index) server 340 of Figure 3 at page 13, line 10 through page 14, line 2 of the application as filed. Accordingly, the Examiner respectfully requests that the examiner reconsider and withdraw the rejection of Claims 1, 5-10, 12-14, 16, and 17 under 35 U.S.C. §112, first paragraph.

On page 3 of the Office Action, the Examiner rejected Claims 1, 5-10, 12-14, 16 and 17 under 35 U.S.C. §112, second paragraph, as being incomplete for omitting essential steps, specifically the steps of indexing. Applicant has amended Claims 1 and 7 to state "for locating assets stored on a storage device for indexing". The step of indexing is clearly listed in each claim and is not omitted. Support for the indexing step is contained in the specification as described above.

Rejections under 35 U.S.C. §103

On page 3 of the Office Action, the Examiner rejected Claims 1, 5-10, 12-14, 16 and 17 under 35 U.S.C. §103(a) as being unpatentable over U.S. Patent No. 5,963,958 to Cottrill in view of U.S. Patent No. 4,931,928 to Greenfield.

The Present Invention

The present invention is an asset locator for locating software assets, code assets and the like that are stored in code repositories used by software designers. A crawl process is performed on a storage device on which assets are stored to identify the assets. Asset-specific parameters related to the stored assets are identified, and the assets are then analyzed based upon these parameters. Specifically, Claim 1 recites "performing a crawl process on said storage device to identify stored assets; identifying the asset type of, and asset-specific parameters related to, said stored assets, said asset-specific parameters comprising languages in which each code asset is written; analyzing said stored assets based on said identified asset-specific parameters" (Claim 1, lines 3-6). After analysis, textual and semantic information is extracted from the stored assets and then the extracted textual and semantic information is stored and indexed for retrieval. Claim 1 further recites "extracting textual and semantic information from said stored assets, said semantic information including semantic information specific to the asset type of each stored asset; and storing and indexing said extracted textual and semantic information for retrieval" (Claim 1, lines 7-9).

The present invention has particular application in a software-development environment where the stored code assets may number in the millions and may be written in diverse languages such as, for example, Java, C/C++, COBOL, HTML, and/or XML.

In a preferred embodiment, a series of data analyzers that are specific to each type of data contained in the code repositories (e.g., a Java analyzer, a C/C++ analyzer, a COBOL analyzer, an HTML analyzer, and/or an XML analyzer) are integrated into the system so that they can be used to search the code repositories using particular attributes specific to the semantics of a particular language used to code the asset. The results of the crawling are stored and indexed in a database. Ordinary keyword searching can then be used with the system, either independently or combined with the attribute-specific semantic searching, to search the database.

U.S. Patent No. 5,963,958 to Cottrill

U.S. Patent No. 5,963,958 to Cottrill ("Cottrill") teaches a computer network with a server computer system including a database and an object definition generator. The database is used to store a plurality of objects for easy access by a user of the system. After searching the database, a user selects an object and the object definition generator generates output code including an object definition for the selected object in a specific programming language. The Examiner acknowledges that Cottrill fails to disclose performing a crawl process on a storage device to identify stored assets, and storing and indexing extracted textual and semantic information for retrieval.

U.S. Patent No. 4,931,928 to Greenfeld

U.S. Patent No. 4,931,928 to Greenfeld ("Greenfeld") teaches an apparatus for source code analysis. The apparatus extracts programming semantics information from an inputted source code. An analysis component employs a database interface which enables extracted programming semantics information to be placed in a user desired database for subsequent recall by a desired query system. The Examiner relies on Greenfeld for an alleged teaching of performing a crawl process on a storage device to identify stored assets and storing and indexing extracted textual and semantic information for retrieval.

The Examiner Has Not Established a *Prima Facie* Case of Obviousness

As set forth in the MPEP:

> To establish a *prima facie* case of obviousness, there must be some suggestion or motivation, either in the references themselves or in the knowledge generally available to one of ordinary skilled in the art, to modify the reference or to combine reference teachings.
> MPEP 2143

As noted above, the present claimed invention specifically claims a method and system for locating assets from among stored assets of diverse types, and in particular, diverse types of code assets. The claimed invention includes the performance of a crawl process on a storage device on which diverse types of code assets are stored, so as to identify the stored assets. The various asset type and asset specific parameters are then identified before the stored assets are analyzed based on the asset-specific parameters.

Cottrill, as acknowledged by the Examiner, contains no teaching or suggestion of the performance of a crawl process of any kind, nor is Cottrill concerned with storing and indexing extracted textual and semantic information for later retrieval. Cottrill merely teaches a system where coded objects are stored in a user-searchable database, and upon selection by a user, an object definition can be generated for an object. This generation of an object definition allows a software application to access the selected object where prior to the object definition generation, the software application would be unable to access the object (e.g., the stored object is written in a different programming language than the application). Cottrill, however, is never concerned with indexing and locating stored assets on a storage device. The database of Cottrill is already constructed (and no instructions are given or suggested to as to how the database was constructed), and there is no teaching of identifying asset types and asset-specific parameters relating to the stored objects on a storage device as is specifically claimed in the present invention.

The Examiner relies on Greenfeld for an asserted teaching of performing a crawl process on a storage device to identify stored assets and storing and indexing extracted textual and semantic information for retrieval. Greenfeld, by way of contrast, contains no teaching or suggestion of performing a crawl on a storage device to identify stored assets. The text of Greenfeld cited by the Examiner contains no teaching or suggestion as to how the source file that is analyzed is located; it presupposes that a user of the Greenfeld system already knows which source file they wish to analyze. The user simply analyzes a source file of interest. With no teaching of performing a crawl on a storage device to identify stored assets, no motivation can be found to instruct one of ordinary skill in the art to modify Cottrill as suggested by the Examiner to achieve the present claimed invention.

The present claimed invention, on the other hand, actually locates code assets of diverse types by performing a crawl process on a storage device, and then analyzes any stored assets after identifying the types of assets and parameters specific to each type. Since these elements are expressly claimed in independent Claim 1 (as well as independent Claim 7), and since Cottrill and Greenfeld, whether considered alone or in combination, contains neither a teaching nor a suggestion of these claimed features, the rejection of Claims 1, 5-10, 12-14, 16 and 17 under 35 U.S.C. §103(a) based on Cottrill in view of Greenfeld is improper. Accordingly, the Examiner is respectfully requested to reconsider and withdraw the rejection of Claims 1, 5-10, 12-14, 16 and 17 under 35 U.S.C. §103(a)

<u>Conclusion</u>

The claimed invention is neither taught nor suggested by Cottrill and Greenfeld, either alone or in combination. Accordingly, reconsideration of the present application, and withdrawal of the rejections is respectfully requested.

The Commissioner is hereby authorized to charge any fees associated with this communication to Deposit Account No. 09-0461.

Respectfully submitted,

[illegible]

October 5, 2006
Date

John R. Brancolini
Registration No. 57,218
SYNNESTVEDT & LECHNER LLP
Suite 2600 Aramark Tower
1101 Market Street
Philadelphia, PA 19107
Telephone: (215) 923-4466
Facsimile: (215) 923-2189